Программирование

Python

Работа с сетью

Черновик

Кафедра ИВТ и ПМ
ЗабГУ

2018

План

Прошлые темы

HTTP-запросы

- HTTP
- requests
- Данные для машинной обработки
- API в веб-приложениях
- HTML в Jupyter

Веб-фреимворк

Outline

Прошлые темы

- Как установить python пакет?
- Для чего используются типы bytes и bytearray?
- Как перевести данные хранящиеся в bytes в строковый формат?
- Что такое словарь (dict)?
- Что такое срез?
- Что делают следующие строчки кода?

Outline

Outline

HTTP

HTTP (HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи произвольных данных.

Протокол передачи данных — набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами.

Интерфейс - совокупность средств, методов и правил взаимодействия (управления, контроля и т.д.) между элементами системы

HTTP

- Во время взаимодействия двух машин по протоколу HTTP одна выступает **сервером**, другая - **клиентом**.
- Клиент посылает **запросы (request)** серверу. Запрос, может требовать от сервера HTML страницу или любой конкретный файл.
- Запрос должен содержать конкретную информации о запрашиваемом ресурсе: адрес сервера, адрес запрашиваемого файла, название запрашиваемой программы

```
GET /wiki/страница HTTP/1.1
Host: ru.wikipedia.org
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9b5) Gecko/20
Accept: text/html
Connection: close
(пустая строка)
```

HTTP

- В ответ на запрос сервер отправляет **ответ (response)**
- Ответ HTTP сервера состоит из заголовков (полей со значениями) и данных непосредственно. Также ответ содержит код, говорящий о статусе операции[1].

```
HTTP/1.1 200 OK
Date: Wed, 11 Feb 2009 11:20:59 GMT
Server: Apache
X-Powered-By: PHP/5.2.4-2ubuntu5wm1
Last-Modified: Wed, 11 Feb 2009 11:20:59 GMT
Content-Language: ru
Content-Type: text/html; charset=utf-8
Content-Length: 1234
Connection: close
(пустая строка)
(запрошенная страница в HTML)
```

- Данные (если есть) всегда располагаются в конце ответа. В примере это html-страница.

[1] Список кодов состояния HTTP

Outline

requests

Для Python существуют несколько популярных пакетов, работающих с протоколом HTTP, среди них urllib, urllib2 и **requests**.

Они облегчают составление HTTP запросов таким образом, что нужно указывать лишь URI (идентификатор запрашиваемого ресурса).

Рекомендуется использовать именно requests так как пакет имеет лаконичный интерфейс (API) и широкие возможности.

Загрузка файлов

Загрузка[2] и сохранение файла

```
import requests

url = "https://raw.githubusercontent.com/VetrovSV" \
"/Programming/master/jupiter.jpg"

response = requests.get( url )

if response.status_code == 200:
    f = open('image.jpg', 'wb')

    # поле content имеет тип данных bytes
    f.write( response.content )
    f.close()
```

[2] перед загрузкой стоит проверить доступность файла например в браузере

HTTP заголовки

HTTP заголовки хранятся в поле headers.
Тип данных этого поля имеет много общего с типом данных словарь.

```
r.headers.keys()
```

KeysView('Content-Security-Policy': "default-src 'none'; style-src 'unsafe-inline'; sandbox 'Strict-Transport-Security': 'max-age=31536000', 'X-Content-Type-Options': 'nosniff', 'X-Frame-Options': 'deny', 'X-XSS-Protection': '1; mode=block', 'ETag': '"00127c6f34d28c415c2f459f8ef814f6495b24e6“, 'Content-Type': 'image/jpeg', 'Cache-Control': 'max-age=300', 'X-Geo-Block-List': '', 'X-GitHub-Request-Id': '0996:59C6:16D44A:183AA3:5AED6BBD', 'Content-Length': '477025', 'Accept-Ranges': 'bytes', 'Date': 'Sat, 05 May 2018 08:30:54 GMT', 'Via': '1.1 varnish', 'Connection': 'keep-alive', 'X-Served-By': 'cache-bma7030-BMA', 'X-Cache': 'MISS', 'X-Cache-Hits': '0', 'X-Timer': 'S1525509054.964947,VS0,VE155', 'Vary': 'Authorization,Accept-Encoding', 'Access-Control-Allow-Origin': '*', 'X-Fastly-Request-ID': '3f2df84b03b16e3a3b7cc37de94ac7be93934119', 'Expires': 'Sat, 05 May 2018

HTTP заголовки

Доступ к полям ответа сервера доступен с помощью оператора [], так же как для типа данных *словарь*.
Все поля имеют строковый тип.

```
r.headers['content-type']
# 'image/jpeg'

r.headers['Content-Length']
# '477025'
```

Не все поля http запроса обязательны. Некоторые могут быть использованы в зависимости от настроек сервера.

Outline

Страницы для машинной обработки

Сайты некоторых организаций имеют специальные страницы для с постоянно востребованной и часто меняющейся информацией.

Примерами могут служить курсы валют, расписание рейсов, прогнозы погоды и предупреждения о чрезвычайных ситуациях.

Такая информация может быть представлена не только в виде удобным для восприятия человеком, но и для машинного анализа.

Пример. Курсы валют

Банк России

Центральный банк Российской Федерации

По адресу https://www.cbr-xml-daily.ru/daily_utf8.xml доступны курсы валют, устанавливаемые Центральным банком[3].

Для данного сервиса[4] существуют ограничения: *не более 5 запросов в секунду и 120 запросов в минуту с одного IP, пожалуйста. Если вам надо больше, то вы что-то делаете не так.*

---

[3] подробнее: cbr-xml-daily.ru

[4] подобные ограничения есть почти для всех сервисов, предоставляющих информацию для машинной обработки. Некоторые блокируют запросы с данного IP после недобросовестного использования

Пример. Курсы валют

Страница создана специально для машинной обработки. Её содержание выглядит так[5]:

<?xml version="1.0"encoding="utf-8"?><ValCurs
Date="05.05.2018"name="Foreign Currency Market»<Valute
ID="R01010»<NumCode>036</NumCode><CharCode>AUD</CharCode><Nomi
доллар</Name><Value>47,5526</Value></Valute><Valute
ID="R01020A»<NumCode>944</NumCode><CharCode>AZN</CharCode><Nom
манат</Name><Value>37,1444</Value></Valute><Valute
ID="R01035»<NumCode>826</NumCode><CharCode>GBP</CharCode><Nomi
стерлингов Соединенного
королевства</Name><Value>85,7008</Value></Valute><Valute
ID="R01060»<NumCode>051</NumCode><CharCode>AMD</CharCode><Nom
драмов</Name><Value>13,0111</Value></Valute><Valute
ID="R01235»<NumCode>840</NumCode><CharCode>USD</CharCode><Nomi
США</Name><Value>63,2012</Value></Valute>

[5] это исходный код страницы, здесь представлен XML

Пример. Курсы валют

Выделим запись о курсе Доллара США

<Valute ID="R01235»
<NumCode>840</NumCode>
<CharCode>USD</CharCode>
<Nominal>1</Nominal>
<Name>Доллар США</Name>
<Value>63,2012</Value></Valute>

Можно ограничится простым разбором строки[6], не прибегая к специальным модулям для разбора XML.

Курс вылюты записан после её названия *<Name>Доллар США</Name>* внутри тегов *<Value>* и *</Value>*

[6] что в большинстве случаев может считаться плохой практикой, но для не сложного примера используем именно этот подход

Пример. Курсы валют

```
# получение обменного курса доллара установленного ЦБ РФ
import requests

r = requests.get("https://www.cbr-xml-daily.ru/daily_utf8.xml")
text = r.text

# начало подстроки с курсом
n1 = text.index("Доллар США") + 24
# конец подстроки с курсом
n2 = text.index("<", n1)

exch_rate = float( text[n1:n2].replace(',','.') )
print(exch_rate)
# 63.2012
```

Outline

API

Некоторые интернет-сервисы (веб-приложения) имеют специальный API для взаимодействия с другими программами.

API (программный интерфейс приложения, application programming interface) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах.

API

API делает взаимодействие с интернет-сервисом более гибким. Можно получать конкретные данные, указывая конкретные парметры.

Однако чтобы воспользоваться API часто нужно получать специальный код (API key), который придётся указывать при каждом обращение к сервису.

В некоторых случаях API key доступен после регистрации, в некоторых - только по платной подписке.

OpenWeather API

Пример запроса к openweathermap.org для получения данных о погоде:

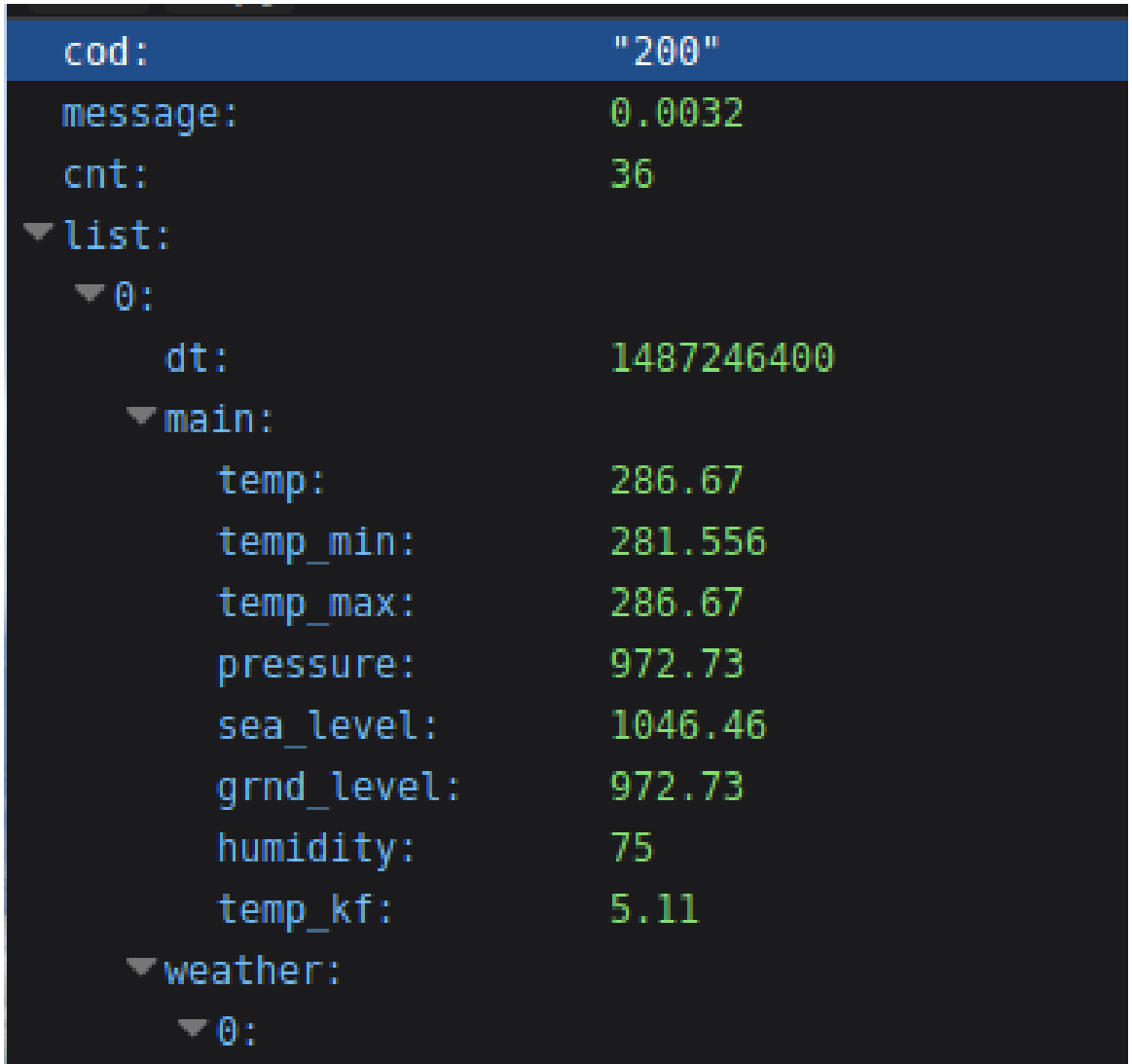
http://samples.openweathermap.org/data/2.5/forecast?q=**Chita,ru**
&appid=**b1b15e88fa797225412429c1c50c122a1**[7]

[7] Этот запрос выдаёт только пример того, как могут выглядеть данные о погоде. Для актуальных данных нужно получить индивидуальный API key

OpenWeather API

OpenWeather в ответ на такой запрос выдаёт текстовые данные в формате JSON.

Браузеры отображают эти данные так:



OpenWeather API

Но на самом деле выглядят они так:


```
{"cod":"200","message":0.0032,"cnt":36,"list":[{"dt":1487246400,"main":
{"temp":286.67,"temp_min":281.556,"temp_max":286.67,"pressure":972.73,"se
[{"id":800,"main":"Clear","description":"clear sky","icon":"01d"}],"cloud
12:00:00"},{"dt":1487257200,"main":
{"temp":285.66,"temp_min":281.821,"temp_max":285.66,"pressure":970.91,"se
[{"id":800,"main":"Clear","description":"clear sky","icon":"01d"}],"cloud
15:00:00"},{"dt":1487268000,"main":
{"temp":277.05,"temp_min":274.498,"temp_max":277.05,"pressure":970.44,"se
[{"id":800,"main":"Clear","description":"clear sky","icon":"01n"}],"cloud
18:00:00"},{"dt":1487278800,"main":
{"temp":272.78,"temp_min":271.503,"temp_max":272.78,"pressure":969.32,"se
[{"id":800,"main":"Clear","description":"clear sky","icon":"01n"}],"cloud
21:00:00"},{"dt":1487289600,"main":
{"temp":273.341,"temp_min":273.341,"temp_max":273.341,"pressure":968.14,"
[{"id":803,"main":"Clouds","description":"broken clouds","icon":"04n"}],"
{"pod":"n"},"dt_txt":"2017-02-17 00:00:00"},{"dt":1487300400,"main":
{"temp":275.568,"temp_min":275.568,"temp_max":275.568,"pressure":966.6,"s
```

Пример. OpenWeather API

Ответ на запрос:


```
{"coord":{"lon":113.5,"lat":52.03},"weather":[{"id":800,"main":"Clear",
"description":"clear sky","icon":"01d"}],"base":"stations","main":
{"temp":14,"pressure":1007,"humidity":26,"temp_min":14,"temp_max":14},
"visibility":10000,"wind":{"speed":1},"clouds":{"all":0},"dt"
:1525586400,"sys":{"type":1,"id":7252,"message":0.0038,"country":"RU",
"sunrise":1525553236,"sunset":1525607944},"id":2025339,
"name":"Chita","cod":200}
```


Сниппет:

```
import requests
url = "http://api.openweathermap.org/data/2.5/weather?"\
    "q=Chita,ru&units=metric&APPID=111111111111"
r = requests.get(url)
text = r.text
n1 = text.index('"temp":') + 7
n2 = text.index(",", n1)
temp = text[n1:n2]
print(temp)
```

Outline

HTML в Jupyter

Часто анализирую результаты запросов приходится просматривать ответы сервера.
Если стоит задача просмотреть выданную сервером HTML страницу то в Jupyter это задача решается следующим образом

```
import requests
from IPython.core.display import display, HTML

url = 'http://habr.ru'
page = requests.get(url)

display(HTML(page.text))
```

В результате в выводе будет показана статичная HTML страница (без изображений).

Outline

Фреймворк

Фреймворк (каркас, framework) — программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Flask

Flask — фреймворк для создания веб-приложений.

Flask - микрофреимворк, т.е. предоставляет только самые базовые возможности[8].

[8] Ещё есть Bottle

Пример веб-приложения, которое показывает «Hello World!»:

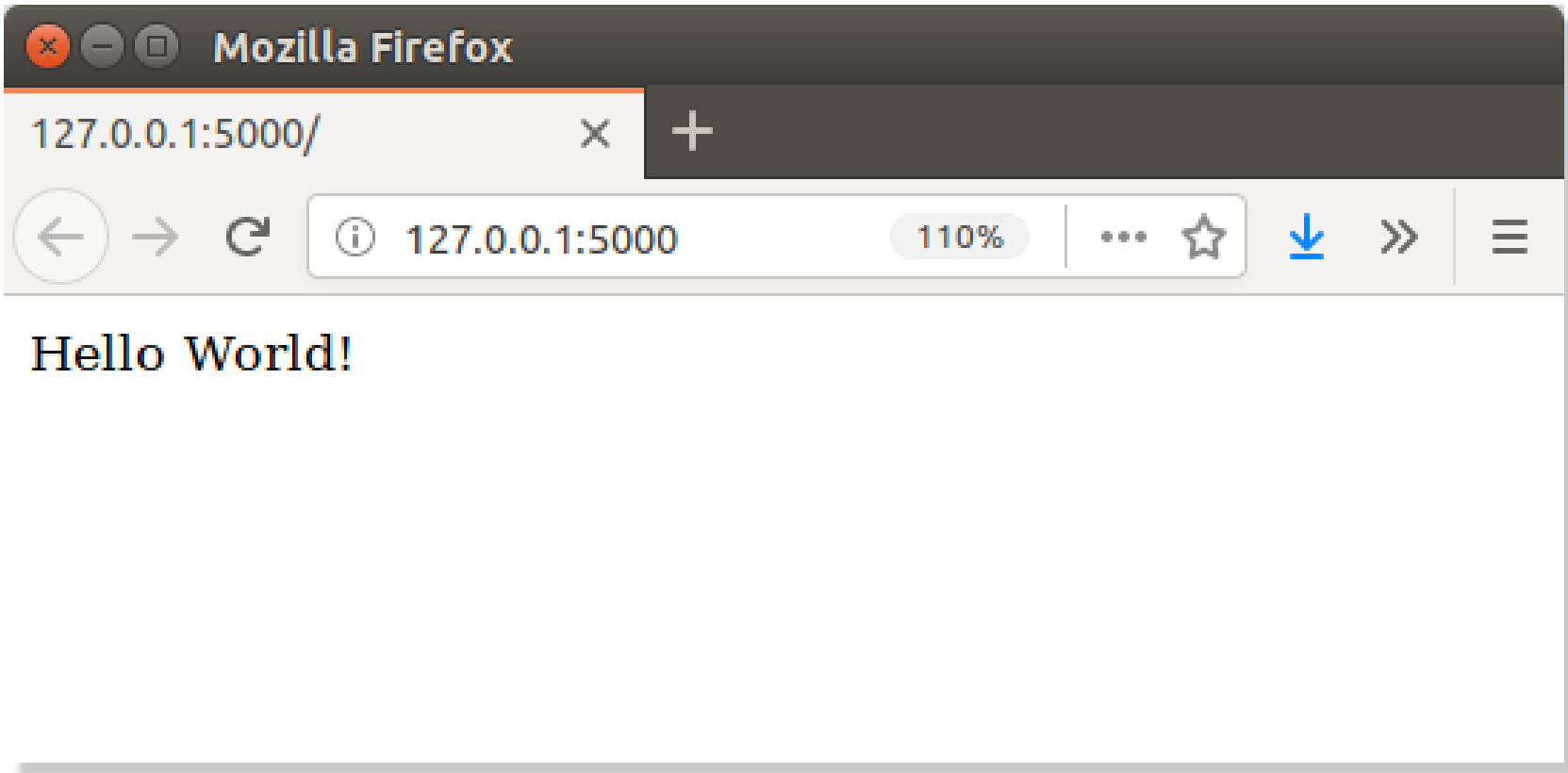
```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

app.run()
```

Будет запущен веб-сервер с единственной страницей:

Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)


Mozilla Firefox
127.0.0.1:5000/
127.0.0.1:5000
110%
Hello World!

Полнофункциональные web-фреймворки

Ссылки и литература

- docs.python-requests.org/en/master - Requests: HTTP for Humans
- http://docs.python-requests.org/en/master/user/authentication/ - requests: Authentication

Ссылки и литература

Ссылка на слайды

github.com/VetrovSV/Programming